Supplemental Material

S1. Gradient of the polynomial approximation of AUC

The gradient of the approximate AUC with respect to the parameter θ is as follows,

$$\frac{\partial AUC^{WMW}(P_\theta,\tau)}{\partial\theta}=\frac{1}{n_0 n_1}\sum_{\mu=0}^{d}\sum_{l=0}^{\mu}\gamma_{\mu l}\left(\frac{\partial s\left({P_\theta}^l,D^\tau\right)}{\partial\theta}v\left({P_\theta}^{\mu-l},D^{!\tau}\right)+s\left({P_\theta}^l,D^\tau\right)\frac{\partial v\left({P_\theta}^{\mu-l},D^{!\tau}\right)}{\partial\theta}\right) \quad \text{(S1)}$$

Note that the calculation of $\partial s\left({P_\theta}^l,D^\tau\right)/\partial\theta$ and $\partial v\left({P_\theta}^{\mu-l},D^{!\tau}\right)/\partial\theta$ is similar, so we only explain one of them. In particular,

$$\frac{\partial s\left({P_\theta}^l,D^\tau\right)}{\partial\theta}=\sum_{i=1}^{L}\frac{\partial[\delta_i^\tau P_\theta(y_i^\tau|\boldsymbol{X})]^l}{\partial\theta} \quad \text{(S2)}$$

Let $Q_i(P_\theta)=[\delta_i^\tau P_\theta(y_i^\tau|x)]^l$, then

$$\frac{\partial s\left({P_\theta}^l,D^\tau\right)}{\partial\theta}=\sum_{i=1}^{L}\left[Q_i'\cdot\frac{\partial P_\theta(y_i^\tau|\boldsymbol{X})}{\partial\theta}\right] \quad \text{(S3)}$$

Where Q_i' is the gradient of Q_i with respect to the marginal probability P_θ.

Since $P_\theta(y_i^\tau|\boldsymbol{X})=\frac{1}{Z(\boldsymbol{X})}\cdot\sum_{\boldsymbol{y}_{1:L}}\left[\delta(y_i=\tau)\cdot\exp\left(\mathbb{F}_{1:L}(\boldsymbol{X},\boldsymbol{y},\theta)\right)\right]$, applying the quotient rule we can compute the gradient of equation (S3) as follows.

$$\frac{\partial s\left({P_\theta}^l,D^\tau\right)}{\partial\theta}=\sum_{i=1}^{L}\left[\frac{1}{Z(\boldsymbol{X})}\cdot Q_i'\cdot\sum_{\boldsymbol{y}_{1:L}}\left[\delta(y_i=\tau)\cdot\frac{\partial\mathbb{F}_{1:L}(\boldsymbol{X},\boldsymbol{y},\theta)}{\partial\theta}\cdot\exp\left(\mathbb{F}_{1:L}(\boldsymbol{X},\boldsymbol{y},\theta)\right)\right]\right]$$

$$-\frac{1}{Z(\boldsymbol{X})}\frac{\partial Z(\boldsymbol{X})}{\partial\theta}\cdot\sum_{i=1}^{L}[Q_i'\cdot P_\theta(y_i^\tau|\boldsymbol{X})] \quad \text{(S4)}$$

The second term in equation (S4) could be calculated efficiently using forward-backward algorithm. For parameter T at position i, the gradient could be calculated as follow.

$$-\sum_{u'}\sum_{u}\frac{\alpha(u',i-1)\cdot\beta(u,i)}{Z(\boldsymbol{X})}\cdot\exp\left(\mathbb{f}_\theta(u',u,\boldsymbol{X},i)\right)\cdot func\cdot\frac{\partial f_\theta(u',u,\boldsymbol{X},i)}{\partial\theta} \quad \text{(S5)}$$

For parameter U at position i, the gradient could be calculated as follows.

$$-\sum_{u}\frac{\alpha(u,i)\cdot\beta(u,i)}{Z(\boldsymbol{X})}\cdot func\cdot\frac{\partial g_\theta(u,\boldsymbol{X},i)}{\partial\theta} \quad \text{(S6)}$$

Where u denotes one label, and

$$func=\sum_{i=1}^{L}[Q_i'\cdot P_\theta(y_i^\tau|\boldsymbol{X})] \quad \text{(S7)}$$

The forward function $\alpha(u,i)$ and backward function $\beta(u,i)$ are defined as follows.

$$\alpha(u,i)=\sum_{\boldsymbol{y}_{1:i}}\delta(y_i=u)\cdot\exp\left(\mathbb{F}_{1:i}(\boldsymbol{X},\boldsymbol{y},\theta)\right) \quad \text{(S8)}$$

$$\beta(u,i)=\sum_{\boldsymbol{y}_{i:L}}\delta(y_i=u)\cdot\exp\left(\mathbb{F}_{i+1:L}(\boldsymbol{X},\boldsymbol{y},\theta)\right) \quad \text{(S9)}$$

They can be calculated by dynamic programming as follows.

$$\alpha(u,i)=\sum_{u'}\alpha(u',i-1)\cdot\exp\left(\mathbb{f}_\theta(u',u,\boldsymbol{X},i)\right) \quad \text{(S10)}$$

$$\beta(u,i)=\sum_{u'}\beta(u',i+1)\cdot\exp\left(\mathbb{f}_\theta(u,u',\boldsymbol{X},i+1)\right) \quad \text{(S11)}$$

The gradient of the inner summation part of the first term in equation (S4) with respect to parameter T at position i could be calculated as follows.

$$\sum_{u'}\sum_{u}\varphi(u',u,i)\cdot\exp\left(\mathbb{f}_\theta(u',u,\boldsymbol{X},i)\right)\cdot\frac{\partial f_\theta(u',u,\boldsymbol{X},i)}{\partial\theta} \quad \text{(S12)}$$

$$\text{Where } \varphi(u',u,i)=Q_i'\cdot\delta(y_i=\tau)\cdot\frac{\alpha(u',i-1)\cdot\beta(u,i)}{Z(\boldsymbol{X})}+\frac{\alpha^\tau(u',i-1)\cdot\beta(u,i)}{Z(\boldsymbol{X})}+\frac{\alpha(u',i-1)\cdot\beta^\tau(u,i)}{Z(\boldsymbol{X})} \quad \text{(S13)}$$

Similarly, the inner summation part of the first term in equation (S4) with respect to parameter U at position i could be calculated as follows

$$\sum_u \phi(u,i) \cdot \frac{\partial g_\theta(u,\boldsymbol{X},i)}{\partial \theta} \tag{S14}$$

$$\text{Where} \quad \phi(u,i) = \frac{\alpha^\tau(u,i)\cdot\beta(u,i)}{Z(\boldsymbol{X})} + \frac{\alpha(u,i)\cdot\beta^\tau(u,i)}{Z(\boldsymbol{X})} \tag{S15}$$

Here we define,

$$\alpha^\tau(u,i) = \sum_{t=1}^{i} \sum_{\boldsymbol{y}_{1:i}} \delta(y_t = \tau \wedge y_i = u) \cdot Q'_t \cdot \exp\left(\mathbb{F}_{1:i}(\boldsymbol{X},\boldsymbol{y},\theta)\right) \tag{S16}$$

$$\beta^\tau(u,i) = \sum_{t=i+1}^{L} \sum_{\boldsymbol{y}_{i:L}} \delta(y_t = \tau \wedge y_i = u) \cdot Q'_t \cdot \exp\left(\mathbb{F}_{i+1:L}(\boldsymbol{X},\boldsymbol{y},\theta)\right) \tag{S17}$$

Like the forward matrix $\alpha(u,i)$ and backward matrix $\beta(u,i)$, $\alpha^\tau(u,i)$ and $\beta^\tau(u,i)$ may also be calculated by dynamic programming. In particular, given the initial conditions $\alpha^\tau(u,1) = Q'_1 \cdot \delta(u = \tau) \cdot \alpha(u,1)$ and $\beta^\tau(u,L) = 0$, $\alpha^\tau(u,i)$ and $\beta^\tau(u,i)$ can be computed by the following recurrences:

$$\alpha^\tau(u,i) = \sum_{u'} [\alpha^\tau(u',i-1) + Q'_i \cdot \delta(u = \tau) \cdot \alpha(u',i-1)] \cdot \exp\left(\mathbb{f}_\theta(u',u,\boldsymbol{X},i)\right) \tag{S18}$$

$$\beta^\tau(u,i) = \sum_{u'} [\beta^\tau(u',i+1) + Q'_{i+1} \cdot \delta(u' = \tau) \cdot \beta(u',i+1)] \cdot \exp\left(\mathbb{f}_\theta(u',u,\boldsymbol{X},i)\right) \tag{S19}$$

Let a and b denote the labels at two adjacent sequence positions, then the gradient of equation (S4) with respect to parameter T is:

$$\frac{\partial s(P_\theta{}^l, D^\tau)}{\partial T_{a,b}} = \sum_{i=1}^{L} \left[\tilde{\varphi}(a,b,i) \cdot \exp\left(\mathbb{f}_\theta(a,b,\boldsymbol{X},i)\right)\right] \tag{S20}$$

Where $\tilde{\varphi}(a,b,i) =$

$$Q'_i \cdot \delta(y_i = \tau) \cdot \frac{\alpha(a,i-1)\cdot\beta(b,i)}{Z(\boldsymbol{X})} + \frac{\alpha^\tau(a,i-1)\cdot\beta(b,i)}{Z(\boldsymbol{X})} + \frac{\alpha(a,i-1)\cdot\beta^\tau(b,i)}{Z(\boldsymbol{X})} - \frac{\alpha(a,i-1)\cdot\beta(b,i)}{Z(\boldsymbol{X})} \cdot func \tag{S21}$$

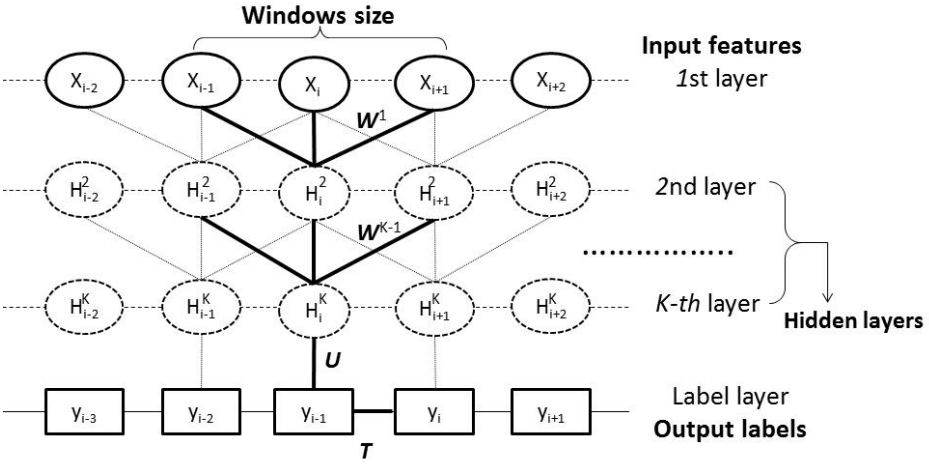
The gradient of equation (S4) with respect to parameter U is:

$$\frac{\partial s(P_\theta{}^l, D^\tau)}{\partial U_{a,h}} = \sum_{i=1}^{L} \left[\tilde{\phi}(a,i) \cdot H_{a,h}(\boldsymbol{X},i,W)\right] \tag{S22}$$

$$\text{Where} \quad \tilde{\phi}(a,i) = \frac{\alpha^\tau(a,i)\cdot\beta(a,i)}{Z(\boldsymbol{X})} + \frac{\alpha(a,i)\cdot\beta^\tau(a,i)}{Z(\boldsymbol{X})} - \frac{\alpha(a,i)\cdot\beta(a,i)}{Z(\boldsymbol{X})} \cdot func \tag{S23}$$

==========

S2. More details about the DeepCNF model



Supplemental Figure 1. Illustration of a DeepCNF. Here i is the position index and X_i the associated input features, H^k represents the k-th hidden layer, and Y is the output label. All the layers from the 1st to the K^{th} (i.e., top layer) form a DCNN with parameter $W^k\{k = 1,2, \dots, K\}$. The K^{th} layer and the label layer form a CRF, in which the parameter U specifies the relationship between the output of the K^{th} layer and the label layer and T the binary relationship between adjacent labels. Windows size is set to 3 only for illustration.

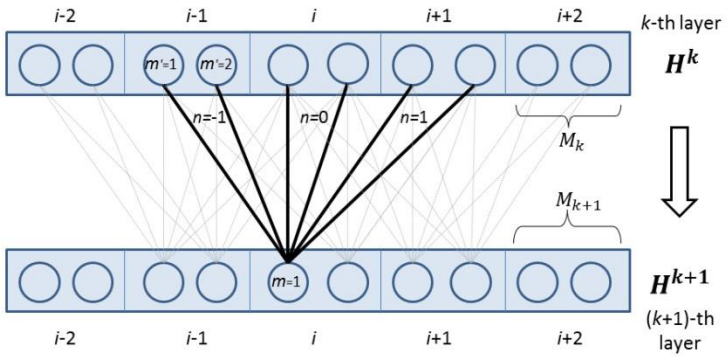
As shown in Supplemental Figure 1, the DeepCNF has three architecture hyper-parameters: (a) the number of neurons at each layer; (b) the window size at each layer; and (c) the number of the hidden layers. We train the model parameters (i.e., *U, T, W*) of DeepCNF simultaneously. We first calculate the gradient for parameter U, T and then for parameter W. Below we explain how to calculate the DeepCNF in a feed-forward way and the gradient by back-propagation.

S2.1 Feed-forward function of DCNN (deep convolutional neural network)

Supplemental Figure 2 shows two adjacent layers of DCNN. Let M_k be the number of neurons for a single position of the k-th layer. Let $X_i(h)$ be the h-th feature at the input layer for residue i and $H_i^k(h)$ denote the output value of the h-th neuron of position i at layer k. When $k = 1$, H^k is actually the input feature $\boldsymbol{X}$. Otherwise, H^k is a matrix of dimension $L \times M_k$. Let $2N_k + 1$ be the window size at the k-th layer. Mathematically, $H_i^k(h)$ is defined as follows.

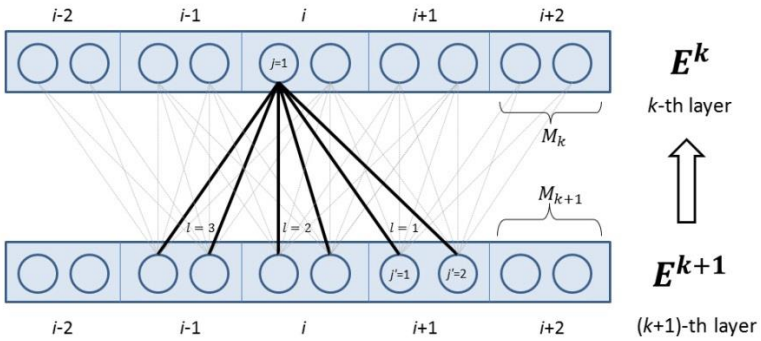
$$\begin{aligned} H_i^k(h) &= X_i(h), && \text{if } k = 1 \\ H_i^{k+1}(h) &= \pi\left(\sum_{n=-N_k}^{N_k}\sum_{h'=1}^{M_k}\left(H_{i+n}^k(h') * W_n^k(h,h')\right)\right), && \text{if } k < K \\ H_h(\boldsymbol{X}, i, W) &= H_i^K(h), && \text{if } k = K \end{aligned} \quad \text{(S24)}$$

Meanwhile, π is the activation function, either the sigmoid (i.e., $(x) = 1/(1 + \exp(-x))$) or the tanh (i.e., $\pi(x) = (1 - \exp(-2x))/(1 + \exp(-2x))$) function. W_n^k $(-N_k \le n \le N_k)$ is a 2D weight matrix for the connections between the neurons of position i at layer k and the neurons of position $i + 1$ at layer $k + 1$. W_n^k is shared by all the positions in the same layer, so it is position-independent. Here h and h' index two neurons at the k-th and $(k + 1)$-th layers, respectively.



Supplemental Figure 2. The feed-forward connection between two adjacent layers of DCNN.

S2.2 Calculation of gradient by back-propagation



Supplemental Figure 3. Illustration of how to calculate the gradient of DCNN from layer $k+1$ to k.

The error function from the CRF part at position i for a certain label u is $E_i(u) = \sum_{\mu=0}^{d}\sum_{l=0}^{\mu}\gamma_{\mu l}\left(\tilde{\phi}_s^l(u,i)\cdot v\left({P_\theta}^{\mu-l}, D^{!\tau}\right) + s\left({P_\theta}^{l}, D^{\tau}\right)\cdot\tilde{\phi}_v^{u-l}(u,i)\right)$, where $\tilde{\phi}_s^l$ and $\tilde{\phi}_v^{u-l}$ are derived according to equation (S23) with respect to function $s\left({P_\theta}^{l}, D^{\tau}\right)$ and $v\left({P_\theta}^{\mu-l}, D^{!\tau}\right)$, respectively. As shown in Supplemental Figure 3, we can calculate the neuron error values as well as the gradients at the k-th layer by back-propagation as follows.

$$E_i^k(h) = \eta(H_i^k(h)) * \sum_u\left[E_i(u) * U_{a,h}\right] \qquad \text{if } k = K$$

$$E_i^k(h) = \eta(H_i^k(h)) * \sum_{n=-N_k}^{N_k}\sum_{h'=1}^{M_{k+1}}\left[E_{i+n}^{k+1}(h') * W_n^k(h',h)\right] \qquad \text{if } k < K \qquad \text{(S25)}$$

Where η is the derivative of the activation function π. In particular, it is $\eta(x) = (1-x)*x$ and $\eta(x) = 1 - x*x$ for the sigmoid and tanh function, respectively. $\boldsymbol{E^k}$ is the neuron error value matrix at the k-th layer, with dimension $L \times M_k$. Finally, the gradient of the parameter W at the k-th layer is:

$$\nabla_{W_n^k(h,h')} = \sum_{i=1}^{L}\left[E_i^{k+1}(h) * H_{i+n}^k(h')\right] \qquad \text{(S26)}$$

==========

S3. Datasets

We use five datasets to train, validate, and evaluate our AUCpreD method. In brief, the Disorder723 dataset [1] is used for determining the model architecture; the UniProt90 dataset [2] is used for training the model parameter; two CASP datasets (i.e., CASP9 [3] and CASP10 [4]) and a recent CAMEO dataset are used for evaluating the model performance. We analyzed the overall order/disorder property of these datasets, as well as the internal disordered regions with different length (shown in Supplemental Table 1). In addition, we use the Human proteome to evaluate the large-scale prediction for different methods.

S3.1 Dataset used for determining the model architecture

We use Disorder723 dataset (http://download.igb.uci.edu/disorder.dataset) to compare different objective functions as well as to determine the model architecture. Disorder723 is a dataset built by Cheng et al. [1] in May 2004, consists of 723 non-redundant chains which span at least 30 amino acids and were solved by X-ray diffraction with a resolution of around 2.5 Å. A ten-fold cross-validation on this Disorder723 dataset was performed. In particular, the original dataset was randomly partitioned into 10 equal-sized subsamples. Among the 10 subsamples, a single subsample is retained as the validation data for testing the model, while the remaining nine are used as training data.

Supplemental Table 1. Number of proteins, order/disorder residues, terminal region disorder residues, and properties of the internal (i.e., non-terminal) disordered regions with different lengths, on the six datasets used in this work.

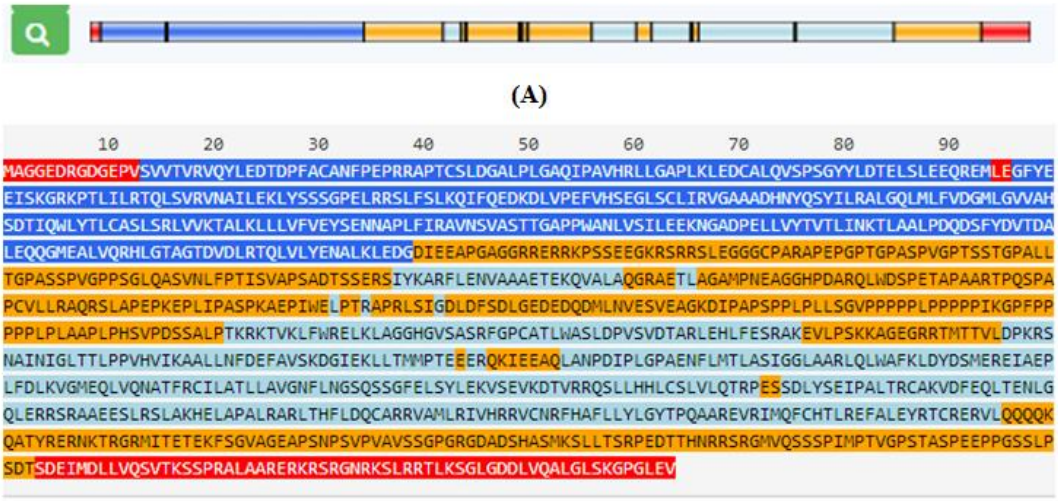
Datasets	**Disorder723**	**UniProt90**	**CASP9**	**CASP10**	**CAMEO**
Overall properties					
Proteins	723	13,800	117	94	229
Ordered residues	201,703	3,374,912	23,656	22,688	56,471
Disordered residues	13,909	178,340	2,679	1,664	5,936
N-terminal disordered residues	4,622	67,391	1,060	814	2,540
C-terminal disordered residues	4,505	55,398	519	258	1,867
Number of internal disordered regions					
1-5	492	2,611	118	73	70
6-15	226	2,730	52	31	56
16-25	45	505	11	6	10
>25	19	291	3	2	13
Length of internal disordered regions					
1-5	964	8,082	272	163	209
6-15	2,083	24,938	494	261	492
16-25	883	9,775	215	113	187
>25	852	12,686	119	55	641

S3.2 Dataset used for training the model parameter

After the model architecture was determined, we further trained our model using UniProt90 dataset [2]. The sequences and the corresponding order/disorder labels from this dataset were downloaded from http://mobidb.bio.unipd.it/lsd. Note that the original UniProt90 dataset were filtered according to the following three criteria: (i) each protein should be released before May-01-2010, which is the starting date of CASP9; (ii) each protein share no more than 25% sequence identity to those sequences from the CASP and CAMEO datasets; and (iii) unannotated amino acid labeled as 'X' are discarded. The remaining non-redundant UniProt90 dataset contains 13,800 proteins with only two labels: 1 for disorder and 0 for order.

S3.3 Dataset for evaluation the model performance

We evaluate the model performance with other state-of-the-art methods on three publicly available dataset: CASP9 (http://predictioncenter.org/download_area/CASP9/targets/casp9.DR_targets.tgz), CASP10 (http://predictioncenter.org/download_area/CASP10/targets/casp10.DR_targets.tgz), and CAMEO of the recent one year (http://www.cameo3d.org/sp/1-year/) (from 2014-09-16 to 2015-09-16). CASP9 dataset contains 117 sequences, CASP10 contains 94, and CAMEO of the recent one year has 229 proteins. Note that we merge CASP9 and CASP10 into the CASP dataset. Among the CAMEO dataset, we only take hard targets according to the official definition.



Supplemental Figure 4. An example of MobiDB order/disorder annotation and prediction result for one Human protein (UniProt entry Q9Y613). Overview **(A)** and detailed **(B)** order/disorder annotations and predictions, where color red (blue) indicates disordered (ordered) residues/regions from experimental and/or manual curation, and color orange (cyan) indicates disordered (ordered) residues/regions from prediction. In this work, we only evaluate the performance of different methods on experimental and/or manual curated order/disorder regions (i.e., color blue and red).

S3.4 Dataset for large-scale prediction

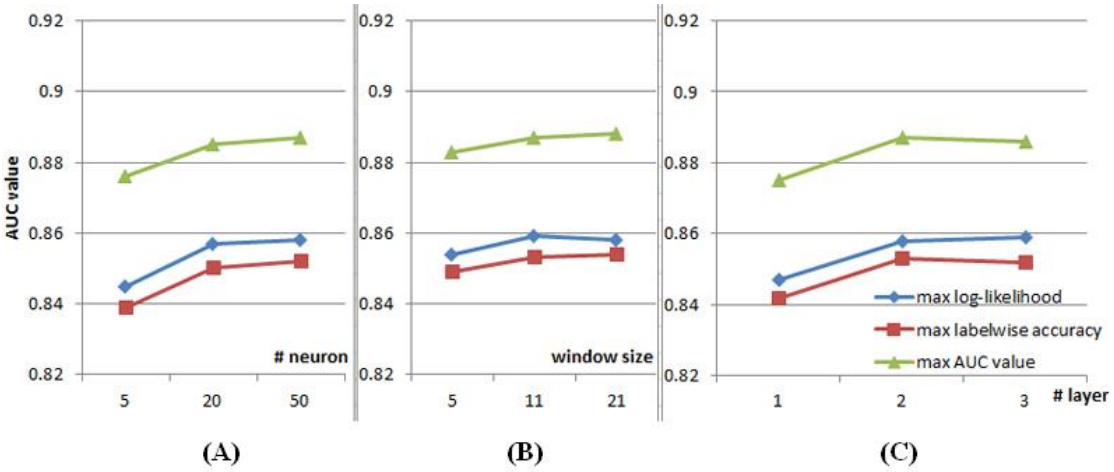
The large-scale prediction across entire Human proteome is an important application of disorder prediction [5]. Here we derive each Human protein with UniProt entry from MobiDB (http://mobidb.bio.unipd.it/). Since the longer the protein length, the more time required to generate the evolution information for prediction. We then restricted to protein of length less than 1700 amino acids, which results in 19,385 valid

Human proteins. Note that MobiDB is designed in three layers in order of quality: (i) manual curation from the DisProt database [6], (ii) experimental PDB information, and (iii) predictions [2]. In order to evaluate and compare the prediction results of different methods, we only take the manual curation and experimental PDB information for order/disorder regions. Among these regions, 1,066,878 are curated as order and 97,815 as disorder. We show one example in Supplemental Figure 4.

==========

S4. Determining the DCNN architecture hyper-parameters

The architecture of the DCNN in DeepCNF model is mainly determined by the following 3 factors (see Supplemental Figure 1): (i) the number of hidden layers; (ii) the number of different neurons at each layer; and (iii) the window size at each layer. We also compared three different methods for training the DeepCNF model (sequence profile not used): maximum likelihood, maximum labelwise accuracy, and maximum AUC. We conduct 10-fold cross-validation for each possible DCNN architecture and each training method. To simplify the analysis, all the hidden layers have the same number of neurons and the same windows size. As shown in Supplemental Figure 5, regardless of the architecture, the AUC-maximization method greatly outperforms the other two in terms of the AUC value. Our model has almost peak performance when it has 2 hidden layers, 50 different hidden neurons at each layer, and windows size set to 11. Further increasing the number of layers and the windows size does not result in significant improvement in AUC, regardless of the training method.



Supplemental Figure 5. Dependency of AUC with respect to the architecture of DCNN. **(A)** the number of neurons, **(B)** window size, and **(C)** the number of hidden layers. Three different training methods: maximum likelihood (blue), maximum labelwise accuracy (red) and maximum AUC (green).

==========

S5. Performance on long disorder regions

To further illustrate the advantage of our method, we examine the prediction accuracy of long disorder regions. As shown in Supplemental Table 2, on CAMEO targets, our method AUCpreD^{p} exceeds all the others in terms of Mcc, AUC^{pr} and AUC on the disorder regions of at least 10 and 20 residues, respectively. The per-residue Mcc obtained by AUCpreD^{p} are 0.46 and 0.40, respectively, higher than the profile-based DisoPred3^{A} (0.39 and 0.34, respectively) and the template-based DisoPred3^{T} (0.41 and 0.35, respectively). Again, without using sequence profile, AUCpreD^{a} is comparable to DisoPred3^{T} in terms of Mcc and AUC.

Supplemental Table 2. Per-residue performance on the CAMEO targets on long disorder regions.

Method	**Longer than 10 residues**						**Longer than 20 residues**					
	Acc	**Sens**	**Spec**	**Mcc**	**AUC^{pr}**	**AUC**	**Acc**	**Sens**	**Spec**	**Mcc**	**AUC^{pr}**	**AUC**
	Predictors using sequence or template information											
AUCpreD^{p}	**0.74**	0.51	**0.97**	**0.46**	**0.49**	**0.86**	**0.71**	0.46	**0.96**	**0.40**	**0.41**	**0.84**
DeepCNF-D^{p}	0.70	0.44	0.95	0.40	0.41	0.83	0.69	0.43	0.95	0.35	0.33	0.81
Espritz^{p}	0.73	0.56	0.89	0.34	0.39	0.79	0.70	0.52	0.89	0.29	0.32	0.77
DNdisorder	0.72	**0.57**	0.88	0.33	0.38	0.77	0.70	**0.53**	0.88	0.28	0.30	0.76
DisoPred3^{A}	0.70	0.45	0.95	0.39	0.40	0.83	0.68	0.41	0.95	0.34	0.33	0.81
DisoPred3^{T}	0.69	0.42	0.96	0.41	0.42	0.82	0.67	0.38	0.96	0.35	0.35	0.80
PrDOS-CNF	0.72	0.47	0.97	0.43	0.43	0.85	0.69	0.43	0.95	0.36	0.36	0.82
	Predictors without using sequence profile information											
AUCpreD^{a}	0.71	0.47	**0.95**	**0.41**	**0.43**	**0.83**	0.70	0.43	**0.95**	**0.36**	**0.35**	**0.81**
DeepCNF-D^{a}	0.68	0.42	0.94	0.35	0.37	0.80	0.67	0.39	0.95	0.31	0.29	0.76
Espritz^{a}	**0.73**	**0.60**	0.87	0.33	0.36	0.79	**0.71**	**0.56**	0.87	0.28	0.31	0.77
IUpred^{S}	0.68	0.43	0.93	0.32	0.33	0.78	0.66	0.40	0.93	0.28	0.25	0.77
IUpred^{L}	0.67	0.42	0.92	0.30	0.29	0.75	0.67	0.42	0.92	0.28	0.25	0.75

==========

S6. Performance on terminal and internal disorder regions

We evaluated the performance of all the predictors on terminal and internal disorder regions. We say a residue is in a terminal region if and only if it is within 10 positions of a protein termini. As shown in Supplemental Table 3, on CAMEO targets, AUCpreD^{p} exceeds all the others in terms of Mcc and AUC on both terminal and internal regions. For internal regions, AUCpreD^{p} also has the best AUC^{pr}. In particular, AUCpreD^{p} has much better Mcc than the profile-based DeepCNF-D^{p}, DisoPred3^{A}, DNdisorder and Espritz^{p}. AUCpreD^{a} has comparable AUC and MCC to DisoPred3^{A} and DisoPred3^{T}.

Supplemental Table 3. Per-residue performance on CAMEO targets on terminal and internal regions.

Method	Terminal protein regions						Internal protein regions					
	Acc	**Sens**	**Spec**	**Mcc**	**AUC^{pr}**	**AUC**	**Acc**	**Sens**	**Spec**	**Mcc**	**AUC^{pr}**	**AUC**
	Predictors using sequence profile or template											
AUCpreD^{p}	**0.75**	0.83	0.68	**0.51**	0.72	**0.85**	**0.68**	0.38	**0.98**	**0.39**	**0.38**	**0.84**
DeepCNF-D^{p}	0.72	0.71	0.72	0.45	0.70	0.82	0.66	0.35	0.97	0.35	0.31	0.79
Espritz^{p}	0.62	**0.97**	0.27	0.30	**0.74**	0.81	0.67	**0.41**	0.92	0.29	0.29	0.75
DNdisorder	0.67	0.83	0.52	0.31	0.69	0.78	0.64	0.34	0.93	0.27	0.27	0.72
DisoPred3^{A}	0.72	0.74	0.70	0.43	0.68	0.79	0.65	0.33	0.96	0.33	0.32	0.80
DisoPred3^{T}	0.73	0.73	**0.74**	0.46	0.69	0.80	0.64	0.30	0.98	0.34	0.34	0.79
PrDOS-CNF	0.73	0.75	0.71	0.45	0.68	0.81	0.66	0.35	0.97	0.36	0.35	0.80
	Predictors without using sequence profile											
AUCpreD^{a}	**0.74**	0.85	0.63	**0.44**	0.71	**0.82**	0.65	0.32	**0.97**	**0.35**	**0.35**	**0.80**
DeepCNF-D^{a}	0.69	0.72	0.67	0.37	0.66	0.79	0.63	0.31	0.95	0.30	0.27	0.75
Espritz^{a}	0.59	**0.98**	0.19	0.25	**0.74**	0.81	**0.68**	**0.45**	0.90	0.28	0.28	0.74
IUpred^{S}	0.67	0.87	0.47	0.34	0.65	0.76	0.62	0.28	0.95	0.25	0.26	0.74
IUpred^{L}	0.64	0.48	**0.80**	0.30	0.57	0.69	0.63	0.33	0.93	0.24	0.23	0.71

==========

Supplemental Reference

1. Cheng, J., Sweredoski, M.J., Baldi, P.: Accurate prediction of protein disordered regions by mining protein structure data. Data mining and knowledge discovery 11, 213-222 (2005)
2. Di Domenico, T., Walsh, I., Martin, A.J., Tosatto, S.C.: MobiDB: a comprehensive database of intrinsic protein disorder annotations. Bioinformatics 28, 2080-2081 (2012)
3. Monastyrskyy, B., Fidelis, K., Moult, J., Tramontano, A., Kryshtafovych, A.: Evaluation of disorder predictions in CASP9. Proteins: Structure, Function, and Bioinformatics 79, 107-118 (2011)
4. Monastyrskyy, B., Kryshtafovych, A., Moult, J., Tramontano, A., Fidelis, K.: Assessment of protein disorder region predictions in CASP10. Proteins: Structure, Function, and Bioinformatics 82, 127-137 (2014)
5. Walsh, I., Martin, A.J., Di Domenico, T., Tosatto, S.C.: ESpritz: accurate and fast prediction of protein disorder. Bioinformatics 28, 503-509 (2012)
6. Sickmeier, M., Hamilton, J.A., LeGall, T., Vacic, V., Cortese, M.S., Tantos, A., Szabo, B., Tompa, P., Chen, J., Uversky, V.N.: DisProt: the database of disordered proteins. Nucleic acids research 35, D786-D793 (2007)